Exhibit H-2

At a step 331, the file server 110 performs a full volume copying operation, as described with regard to the flow point 310 through the flow point 320. The volume copying operation is performed for a full copy of the file system 114.

- o If the function to be performed is full mirroring, the file server 110 performs the full volume copying operation to disk as the target destination file system 120.

- o If the function to be performed is incremental mirroring, the file server 110 performs the full volume copying operation to tape as the target destination file system 120.

At a step 332, the file server 110 sets a mirroring timer for incremental update for the volume mirroring operation.

At a step 333, the mirroring timer is hit, and the file server 110 begins the incremental update for the volume mirroring operation.

At a step 334, the file server 110 performs an incremental volume copying operation, as described with regard to the flow point 310 through the flow point 320. The volume copying operation is performed for an incremental upgrade of the file system 114.

The incremental volume copying operation is performed with disk as the target destination file system 120.

- If the initial full volume copying operation was performed to disk, the destination file system 120 increments its copy of the file system 114 to include the incremental storage image 220.

- If the initial full volume copying operation was performed to tape, the destination file system 120 records the incremental storage image 220 and integrates it into an incremental mirror data structure, as described above, for possibly later incrementing its copy of the file system 114.

At a step 335, the file server 110 copies the image stream 230 to the target destination file system 120. The method 300 returns to the step 332, at which step the file server 110 resets the mirroring timer, and the method 300 continues.

When the destination file system 120 receives the image stream 230, it records the storage blocks 115 in that image stream 230 similar to the process of volume copying, as described with regard to the step 315.

If the method 300 is halted (by an operator command or otherwise), the method 300 completes at the flow point 340.

At a flow point 340, the file server 110 has completed the volume mirroring operation.

Reintegration of Incremental Mirror

At a flow point 370, the file server 110 is ready to restore a file system from the base storage image 220 and the incremental mirror data structure.

At a step 371, the file server 110 reads the base storage image 220 into its file system.

At a step 372, the file server 110 reads the incremental mirror data structure into its file system and uses that data structure to update the base storage image 220.

At a step 373, the file server 110 remounts the file system that was updated using the incremental mirror data structure.

At a flow point 380, the file server 110 is ready to continue operations with the file system restored from the base storage image 220 and the incremental mirror data structure.

Volume Replication

At a flow point 350, the file server 110 is ready to perform a volume replication operation.

At a step 351, the destination file system 120 initiates the volume replication operation. The destination file system 120 sends an indicator of its newest master snapshot 210 to the file server 110, and requests the file server 110 to perform the volume replication operation.

At a step 352, the file server 110 determines if it needs to perform a volume replication operation to synchronize with a second file server 140. In this case, the second file server 140 takes the role of the destination file system 120, and initiates the volume replication operation with regard to the first file server 110.

At a step 353, the file server 110 determines its newest master snapshot 210, and its master snapshot 210 corresponding to the master snapshot 210 indicated by the destination file system 120.

- If the file server 110 has at least one master snapshot 210 older than the master snapshot 210 indicated by the destination file system 120, it selects the corresponding master snapshot 210 as the newest one of those.

In this case, the method proceeds with the step 354.

- If the file server 110 does not have at least one master snapshot 210 older than the master snapshot 210 indicated by the destination file system 120 (or if the destination file system 120 did not indicate any master snapshot 210), it does not select any master snapshot 210 as a corresponding master snapshot.

In this case, the method proceeds with the step 355.

At a step 354, the file server 110 performs an incremental volume copying operation, responsive to the incremental difference between the selected corresponding master snapshot 210, and the newest master snapshot 210 it has available. The method 300 proceeds with the flow point 360.

At a step 355, the file server 110 performs a full volume copying operation, responsive to the newest master snapshot 210 it has available. The method 300 proceeds with the flow point 360.

At a flow point 360, the file server 110 has completed the volume replication operation. The destination file system 120 updates its master snapshot 210 to correspond to the master snapshot 210 that was used to make the file system transfer from the file server 110.

103.1068.01

Alternative Embodiments

Although preferred embodiments are disclosed herein, many variations are possible which remain within the concept, scope, and spirit of the invention, and these variations would become clear to those skilled in the art after perusal of this application.

103.1068.01

Claims

1. A file system, having a plurality of storage blocks, and including a plurality of bits associated with each one of said plurality of storage blocks, at least one of said plurality of bits identifying whether said one storage block was part of said file system at a time earlier than a current consistent version of said file system.

2. A file system as in claim 1, including a second one of said plurality of bits identifying whether said one storage block was part of said file system at a second time earlier than a current consistent version of said file system

3. A file system as in claim 2, including an element disposed for selecting storage blocks in response to said one bit and said second one bit associated with said selected storage blocks.

4. A file system as in claim 3, including an element disposed for copying said selected storage blocks to a destination.

5. A file system as in claim 4, wherein said destination includes: a tape, a disk, a data structure in a second file system, a set of network messages, or a destination distributed over a plurality of file systems.

103.1068.01

6. A file system as in claim 1, including an element disposed for selecting storage blocks in response to said one bit associated with said selected storage blocks.

7. A file system as in claim 6, including an element disposed for copying said selected storage blocks to a destination.

8. A file system as in claim 7, wherein said destination includes: a tape, a disk, a data structure in a second file system, a set of network messages, or a destination distributed over a plurality of file systems.

9. A file system having a plurality of storage blocks, said file system including a snapshot including a set of member storage blocks selected from said plurality, said member storage blocks forming a consistent file system other than an active file system; said snapshot being disposed as an object in said file system, wherein said file system is responsive to at least one file system request with regard to said snapshot.

10. A file system as in claim 9, including

a shadow snapshot of a set of member storage blocks selected from said plurality, said member storage blocks having formed a consistent file system other than an active file system, with a set of selected member storage blocks removed from said consistent file system; and

103.1068.01

a storage image defined in response to said snapshot and said shadow snapshot, said storage image indicating a set of member storage blocks selected from said plurality.

11. A file system as in claim 9, including a plurality of said snapshots; wherein said plurality of said snapshots are associated with an array of bits, said array having one set of bits for each storage block in said plurality of storage blocks, said set of bits having at least one bit for each said snapshot.

12. A file system as in claim 9, wherein said file system can manipulate said snapshot without having to traverse a hierarchy of file system objects within said snapshot.

13. A file system as in claim 9, wherein said snapshot includes a data structure disposed in a format allowing for a set management operation to be performed efficiently.

14. A file system as in claim 9, wherein said snapshot includes an array of bits, said array having one bit for each storage block in said plurality.

15. A file system as in claim 9, including

a plurality of said snapshots; and

a storage image determined in response to said plurality of snapshots;

said storage image defining a second set of member storage blocks selected from said plurality.

16. A file system as in claim 15, wherein said storage image is a result of a set management operation on said set of member storage blocks for said snapshot.

17. A file system as in claim 9, wherein said snapshot includes a data structure disposed in a format allowing for a set management operation to be performed in O(n) time or less, where n is a number of storage blocks in said plurality, without reading any contents of said storage blocks in said plurality.

18. A file system as in claim 17, wherein said set management operation is a logical sum or difference.

19. A file system as in claim 9, wherein said snapshot includes a data structure identifying which storage blocks in said plurality are member storage blocks of said snapshot.

20. A file system as in claim 19, wherein said data structure uses no more than 1/100th of an amount of storage required by said storage blocks in said plurality.

21. A file system as in claim 19, wherein said data structure uses no more than four bytes per storage block in said plurality.

22. A method to be performed in a file system, said file system having a plurality of storage blocks, said method including steps for

defining a storage image of a set of member storage blocks selected from said plurality, said storage image being formed based on a set of member storage blocks forming a consistent file system other than an active file system; and

forming an image stream of a sequence of member storage blocks selected from said storage image.

23. A method as in claim 22, including steps for associating a block location with each one of said sequence.

24. A method as in claim 22, further including steps for reconstructing a file system based on said image stream.

25. A method as in claim 22, including repeating said defining step at periodic intervals.

26. Apparatus including

a file system including a plurality of snapshots thereof, each representing an associated consistent state at an associated selected time; and

each said snapshot including an indication of a set of storage blocks in said associated consistent state, said indication being recorded in at least one storage block in said associated consistent state.

27. In a file system having a plurality of storage blocks, a data structure including

a first snapshot of a set of member storage blocks selected from said plurality, said member storage blocks forming a consistent file system other than an active file system;

said first snapshot being represented as an object in said file system and having a set of storage blocks for recording said first snapshot;

whereby copying said member storage blocks in said first snapshot has the property of preserving at least one snapshot recorded in said file system at a time of said first snapshot.

28. A data structure as in claim 27, including

a second snapshot of a set of member storage blocks selected from said plurality, said member storage blocks forming a consistent file system other than an active file system;

said second snapshot being represented as an object in said file system and having a set of storage blocks for recording said second snapshot;

whereby copying said member storage blocks in said second snapshot has the property of preserving at least one snapshot recorded in said file system at a time of said second snapshot.

29. A data structure as in claim 27, including

an image stream including a set of storage blocks including both said first snapshot and said second snapshot;

whereby copying said member storage blocks in said image stream has the property of preserving both said first snapshot and said second snapshot.

30. In a file system having a plurality of storage blocks, a data structure including

a snapshot of a set of member storage blocks selected from said plurality, said member storage blocks forming a consistent file system other than an active file system;

said snapshot being represented as an object in said file system and having a set of storage blocks for recording said snapshot;

whereby a backup and restore operation on said file system has the property of preserving said snapshot within said file system.

31. In a file system having a plurality of storage blocks, a data structure including

a storage image of a set of member storage blocks selected from said plurality;

said storage image being formed based on a set of member storage blocks forming a consistent file system other than an active file system.

32. A data structure as in claim 31, including

a first storage image indicating a set of member storage blocks forming a consistent file system; and

a sequence of incremental storage images, each having a predecessor, at least one of said predecessors being said first storage image;

wherein a logical sum of said set of storage images includes at least one complete snapshot.

33. A data structure as in claim 31, wherein said storage image indicates a set of member storage blocks forming a consistent file system.

34. In a file system having a plurality of storage blocks, a data structure stored in said file system, including a shadow snapshot of a set of member storage blocks selected from said plurality, said member storage blocks having formed a consistent file

system other than an active file system, with a set of selected member storage blocks removed from said consistent file system.

35. A data structure as in claim 34, wherein said shadow snapshot uses, in addition to said member storage blocks, no more than 1/100th of an amount of storage required by said storage blocks in said plurality.

36. A data structure as in claim 34, wherein said shadow snapshot is disposed as a single object in said file system, whereby said file system can manipulate said snapshot without having to traverse a hierarchy of file system objects within said snapshot.

37. A data structure as in claim 34, wherein said removed member storage blocks are responsive to completion of a processing operation.

38. A data structure as in claim 37, wherein said processing operation includes a file system operation.

39. A data structure as in claim 37, wherein said processing operation includes reuse of said selected member storage blocks by said file system.

40. A data structure as in claim 34, wherein said shadow snapshot is disposed in a format allowing for a set management operation to be performed in O(n) time or less, where n is a number of storage blocks in said plurality, without reading any contents of said storage blocks in said plurality.

41. In a file system having a plurality of storage blocks, a data structure stored in said file system, including a mark-on-allocate image of a set of member storage blocks selected from said plurality, said member storage blocks having been added to a snapshot that originally formed a consistent file system.

42. A data structure as in claim 41, wherein said mark-on-allocate storage image is disposed as a single object in said file system, whereby said file system can manipulate said snapshot without having to traverse a hierarchy of file system objects within said snapshot.

43. A data structure as in claim 41, wherein said mark-on-allocate image is disposed in a format allowing for a set management operation to be performed efficiently.

44. A data structure as in claim 41, wherein said mark-on-allocate storage image uses no more than 1/100th of an amount of storage required by said storage blocks in said plurality.

45. A data structure as in claim 41, said member storage blocks having been selected responsive to completion of a processing operation.

46. A data structure as in claim 45, wherein said processing operation includes a file system operation.

47. A data structure as in claim 45, wherein said processing operation includes reuse of said selected member storage blocks by said file system.

48. A data structure as in claim 41, wherein said mark-on-allocate image is disposed in a format allowing for a set management operation to be performed in O(n) time or less, where n is a number of storage blocks in said plurality, without reading any contents of said storage blocks in said plurality.

49. In a file system having a plurality of storage blocks, a data structure stored in said file system, including a mark-on-deallocate image of a set of member storage blocks selected from said plurality, said member storage blocks having been removed from a snapshot that originally formed a consistent file system.

50. A data structure as in claim 49, wherein said mark-on-deallocate storage image uses no more than 1/100th of an amount of storage required by said storage blocks in said plurality.

51. A data structure as in claim 49, wherein said mark-on-deallocate image is disposed in a format allowing for a set management operation to be performed in O(n) time or less, where n is a number of storage blocks in said plurality, without reading any contents of said storage blocks in said plurality.

52. A method for recording a plurality of data about a plurality of blocks of data stored in storage means, comprising the steps of:

maintaining a means for recording multiple usage bits per block of said storage means; and

storing, in said means for recording multiple usage bits per block, multiple bits for each of said plurality of said blocks of said storage means, at least one of said multiple bits being indicative of block reusability.

53. A method for recording a plurality of data about a plurality of blocks of stored data, comprising the steps of:

recording multiple usage bits per block of said stored data; and

storing multiple bits for each of said plurality of said blocks of stored data, at least one of said multiple bits being indicative of block reusability.

103.1068.01

Abstract of the Disclosure

The invention provides a method and system for duplicating all or part of a file system while maintaining consistent copies of the file system. The file server maintains a set of snapshots, each indicating a set of storage blocks making up a consistent copy of the file system as it was at a known time. Each snapshot can be used for a purpose other than maintaining the coherency of the file system, such as duplicating or transferring a backup copy of the file system to a destination storage medium. In a preferred embodiment, the snapshots can be manipulated to identify sets of storage blocks in the file system for incremental backup or copying, or to provide a file system backup that is both complete and relatively inexpensive. Also in a preferred embodiment, shadow snapshots can be maintained, with a shadow snapshot including a set of member storage blocks that formed a consistent file system other than an active file system, with a set of selected member storage blocks removed from the consistent file system.

09857487.051001

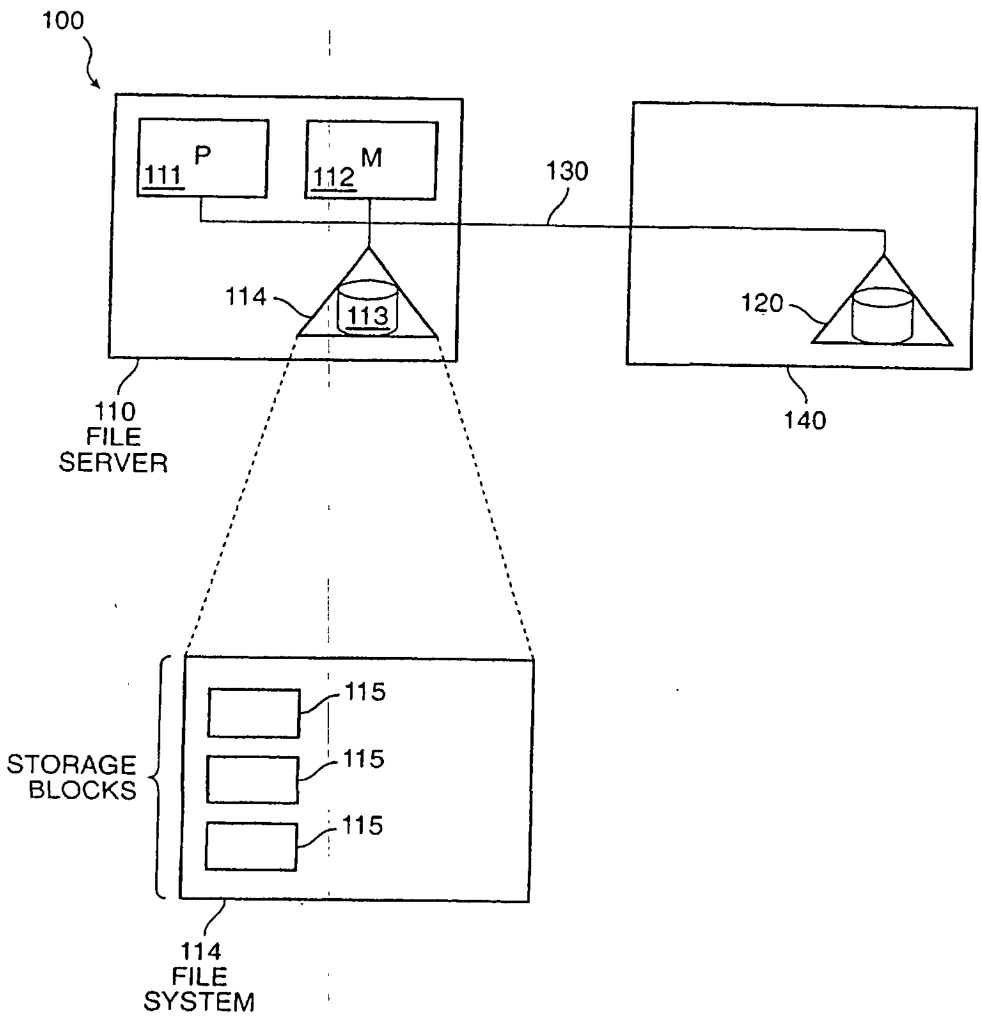
100
111 P
112 M
130
114
113
120
110
FILE
SERVER
140
115
115
115
STORAGE
BLOCKS
114
FILE
SYSTEM

FIG. 1

Title: File System Image Transfer
Inventor: Steven R. KLEIMAN
Docket: 103.1068.01
Sheet No.: Sheet 2 of 4

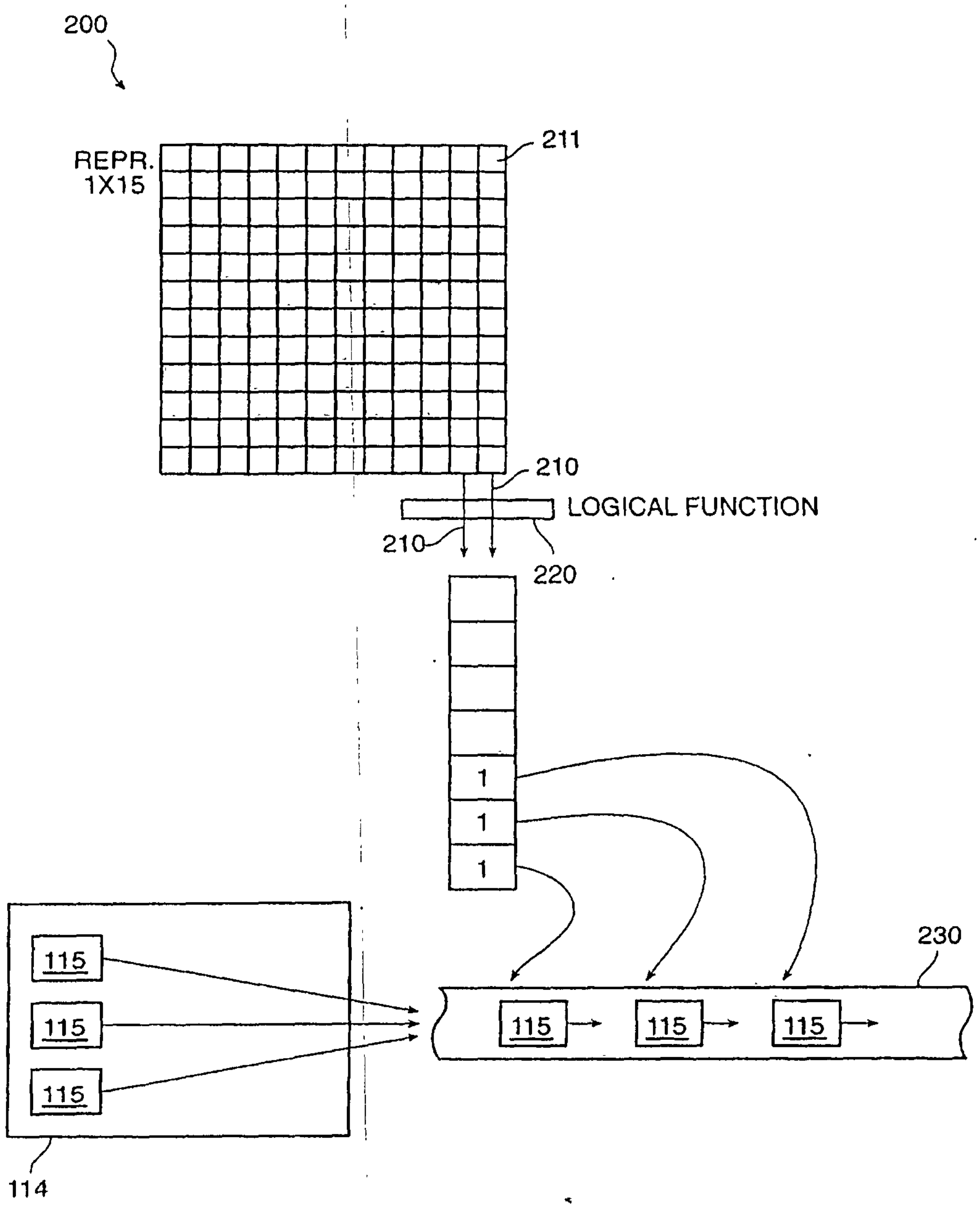

FIG. 2

Inventor: Steven R. KLEIMAN
Docket: 103.1068 01
Sheet No.:: Sheet 3 of 4

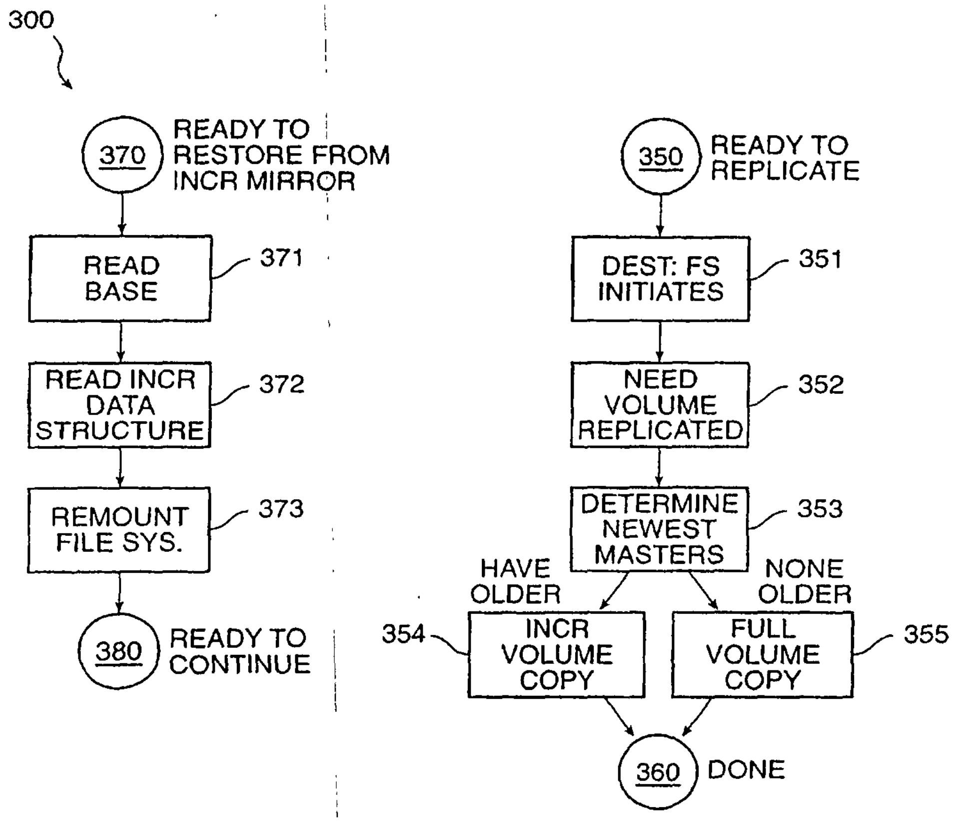

FIG. 3A

Title: File System Image Transfer
Inventor: Steven R. KLEIMAN
Docket: 103.1068.01
Sheet No.:: Sheet 4 of 4

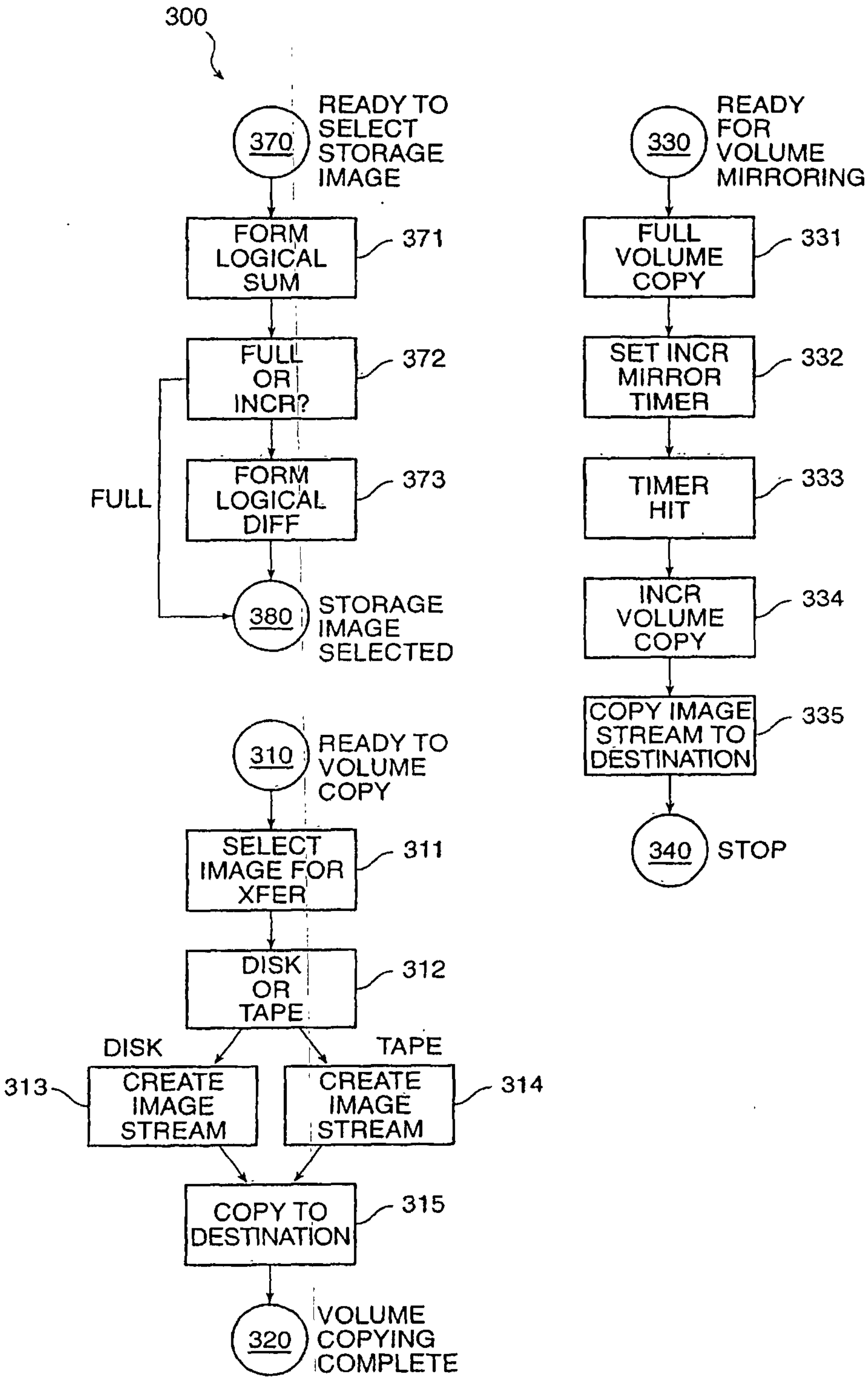

FIG. 3B

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/854,187	05/10/2001	Steven R. Kleiman	103.1068.01	7234

22883 7590 09/29/2004

SWERNOFSKY LAW GROUP PC
P.O. BOX 390013
MOUNTAIN VIEW, CA 94039-0013

EXAMINER	
LE, UYEN T	

ART UNIT	PAPER NUMBER
2171	

DATE MAILED: 09/29/2004

9

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/854,187	KLEIMAN ET AL.	
	Examiner	Art Unit	
	Uyen T. Le	2171	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☐ Responsive to communication(s) filed on _____.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 52-147 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 52-147 is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

Ule

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date 4.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

Claim Rejections - 35 USC § 101

35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

1. Claims 54-61, 99-105, 115-147 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. Note claims 54, 99, 104, 114, 115, 121, 131, 141 merely recite non-functional descriptive material without any practical application. The claimed file system of claim 54 for example recites data stored in blocks associated with a plurality of bits, at least one bit identifying an earlier version of the file system. Although the claim implicitly recites data embodied on a computer-readable medium since bits are digital, the data does not impart functionality to either the data as claimed or to the computer. As such, the claimed invention recites non-functional descriptive material, i.e., mere data. Non-functional descriptive data stored on a computer-readable medium is merely carried on the medium, it is not structurally and functionally interrelated to the medium. Therefore, the claimed subject matter is non-statutory.

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

2. Claims 52-54, 59, 60, 62, 63, 65-73, 76, 77, 79, 80, 87-90, 96, 104, 111, 112, 114-116, 120, 121, 123, 125-129, 131-134, 136-139, 141, 144, 142, 146 are rejected under 35 U.S.C. 102(b) as being anticipated by Raymond A. Lorie "Physical Integrity in a Large Segmented Database", ACM Transactions on Database Systems, Vol.2, No. 1., March 1977, pages 91-104.

Regarding claim 52, Lorie discloses all the claimed subject matter including "maintaining a means for recording multiple usage bits per block of said storage means" when Lorie shows Mod bit and shadow bit (see pages 95-97), and "storing in said means for recording multiple usage bits per block, multiple bits for each of said plurality of said blocks of said storage means, at least one of said multiple bits being indicative of block reusability" when Lorie shows using shadow bits to release slots in the current bit map (see page 99). Clearly the shadow bit is indicative of block reusability.

Regarding claim 53, Lorie discloses all the claimed subject matter including "recording multiple usage bits per block of said stored data" (see pages 95-97, Mod bit and shadow bit), "storing multiple bits for each of said plurality of said blocks of stored data, at least one of said multiple bits being indicative of block reusability" (see page 99). Clearly the shadow bit is indicative of block reusability.

Claim 54 merely reads on the file system of Lorie consisting of a plurality of segments (see the abstract). Segments clearly consist of bits since the file system is digital. The claimed "at least one of said plurality of bits...consistent version of said file system" is met when Lorie shows the Mod bits (see page 95) indicating whether the block has been modified or not.

Regarding claim 59, Lorie discloses an element disposed for selecting storage blocks in response to said one bit and said second bit associated with said selected storage block when Lorie shows that the method selects a free slot (see page 97).

Regarding claim 60, Lorie discloses an element disposed for copying said selected storage blocks to a destination when Lorie shows that a modified segment is copied to a new slot (see page 97).

Regarding claim 62, Lorie discloses all the claimed subject matter including "a file system…with regard to said snapshot" (see the abstract, pages 93, 96, 97).

Claims 63, 65 merely read on the fact that snapshots are formed of member storage blocks which have been added to or removed from the original consistent storage blocks. Lorie teaches such features when Lorie shows periodic backup copies of the database (see page 93).

Regarding claim 66, the claimed shadow snapshot is merely a subset of a snapshot, the member storage blocks no longer forming a consistent file system as defined by applicant in the specification. This feature merely reads on the fact that storage blocks are reusable in the system of Lorie. Clearly, a storage image is defined based on the snapshot and shadow snapshot and indicate a set of member storage blocks selected from a plurality of storage blocks as claimed.

Claims 67, 73, 117 merely read on the fact that the Mod bits keep track of blocks to be copied because they had been modified (see page 95).

Claim 68 merely reads on the fact that the system of Lorie includes multiple snapshots (see page 97).

Regarding claim 69, since the snapshot in the system of Lorie exists as a distinct entity, clearly the system can manipulate the snapshot without having to traverse a hierarchy of file system objects within said snapshot.

Regarding claim 70, since snapshots are formed by adding or removing blocks of data, clearly, the data structure is in a format allowing a set management operation as claimed.

Regarding claim 71, Lorie shows that a snapshot includes an array of bits (see page 96). Clearly, said array has one bit for each storage block since storage blocks are made up of bits.

Regarding claim 72, Lorie discloses a plurality of snapshots (see page 101). Clearly, a storage image is determined in response to said plurality of snapshots and said storage image is defining a second set of member storage blocks selected from said plurality of storage blocks.

Regarding claim 76, since the storage image is formed by adding or removing member storage blocks for said snapshot, clearly said storage image is a result of a set management operation as claimed.

Regarding claim 77, Lorie discloses the claimed "wherein said snapshot...without reading any contents of said storage blocks in said plurality" when Lorie shows the use of shadow bits to release slots in the current bit map (see page 99).

Regarding claim 79, Lorie discloses that each snapshot includes a data structure identifying which storage blocks in said plurality of storage blocks are member storage blocks of said snapshot (see page 99).

Regarding claim 80, since only one bit is required to indicate whether the block is part of a corresponding snapshot, clearly the data structure uses no more than 1/100 of storage amount required by said storage block.

Regarding claim 87, the claimed step of defining a storage image of a set of member storage blocks forming a consistent file system other than an active file system merely reads on the fact that snapshots are formed in the method of Lorie (see page 95). Clearly, snapshots form an image stream of a sequence of member storage blocks as claimed.

Claim 88 merely reads on the fact that each block is associated with a snapshot.

Regarding claim 89, Lorie discloses that the image stream is used to reconstruct the file system when Lorie shows restoring a segment (see page 99).

Claim 90 merely reads on the fact that operations are performed on storage blocks to form snapshots (see pages 94, 95).

Regarding claim 96, the claimed periodic intervals merely read on the fact that snapshots are taken whenever the system advances from one consistent point to another (see page 95).

Regarding claim 104, Lorie discloses a file system including a plurality of snapshots, each representing an associated consistent state at an associated selected time (see page 95). Clearly, an indication is being recorded in at least one storage block in order to show which set of storage blocks form a specific snapshot.

Claim 111 merely reads on the fact that a first snapshot consisting of storage blocks forming a consistent file system other than an active file system is formed in the

system of Lorie and that copying its storage blocks does not alter the snapshot in any way (see page 99).

Regarding claim 112, Lorie discloses a second snapshot when Lorie shows saving a new state (see page 99). Clearly, said second snapshot has a set of member storage blocks forming a consistent file system other than an active file system and is represented as an object in said file system. Claim 112, last paragraph merely reads on the fact that copying said member storage blocks do not alter the snapshot in any way.

Claim 114 merely reads on the fact that a snapshot consisting of storage blocks forming a consistent file system other than an active file system is formed in the system of Lorie (see page 99). Clearly, backup and restore operations do not alter the snapshot in any way.

Claim 115 merely reads on the fact that the system of Lorie produces versions of consistent copies of the file system other than the active file system (see page 99).

Regarding claim 116, Lorie discloses a first storage image indicating a set of member storage blocks forming a consistent file system when Lorie shows the consistent original file system (see the abstract). Lorie discloses a sequence of incremental storage image when Lorie shows saving new states (see page 99). Clearly, each image has a predecessor since snapshots are formed as changes occurs and each snapshot represents a consistency point.

Claim 120 merely reads on the fact that snapshots consist of storage image of a consistency point of the file system.

Regarding claim 121, the claimed shadow snapshot is merely a subset of a snapshot, the member storage blocks no longer forming a consistent file system as defined by applicant in the specification. This feature merely reads on the fact that storage blocks are reusable in the system of Lorie (see page 99).

Regarding claim 123, since only one bit is required to indicate whether the block is part of a corresponding snapshot, clearly the data structure uses no more than 1/100 of storage amount required by said storage block.

Regarding claim 125, since the snapshot in the system of Lorie exists as a distinct entity, clearly the system can manipulate the shadow snapshot without having to traverse a hierarchy of file system objects within said snapshot.

Claims 126, 127, 128 merely read on the fact that member storage blocks are removed from a snapshot in response to an operation on the file system. Lorie clearly shows such a feature when Lorie discloses that blocks are reusable (see page 99).

Regarding claim 129, Lorie discloses the claimed "wherein said snapshot...without reading any contents of said storage blocks in said plurality" when Lorie shows the use of shadow bits to release slots in the current bit map (see page 99).

Claims 131, 141 merely recite components of a snapshot. Clearly, a snapshot consists of mark-on-allocate image or mark-on-deallocate image as claimed because snapshots are consistent points of a file system. Evidently, member storage blocks have been added or removed depending on the consistency of the file system at the time the snapshot was taken.

Regarding claim 132, since the snapshot in the system of Lorie exists as a distinct entity, clearly the system can manipulate the snapshot without having to traverse a hierarchy of file system objects within said snapshot.

Regarding claims 133, 193, since snapshots are formed by adding or removing blocks of data, clearly, the mark-on-allocate image and mark-on-deallocate image are in a format allowing an efficient set management operation as claimed.

Regarding claims 134, 144, since only one bit is required to indicate whether the block is part of a corresponding snapshot, clearly the mark-on-allocate image and mark-on-deallocated image use no more than 1/100 of storage amount required by said storage block.

Claims 136-139 are rejected for the same reasons discussed in claims 129-129 above.

Claim 142 merely reads on the fact that snapshots are removed from the file system. Clearly the mark-on-deallocate image in the system of Lorie is disposed as a single object and the system can manipulate the shadow snapshot without having to traverse a hierarchy of file system objects within said snapshot.

Regarding claim 146, Lorie discloses the claimed "wherein said snapshot...without reading any contents of said storage blocks in said plurality" when Lorie shows the use of shadow bits to release slots in the current bit map (see page 99).

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 55-58, 61, 74, 75, 78, 81-86, 91, 92-95, 97-103,105-110, 113, 118, 119, 124, 130, 135, 140, 145, 147 are rejected under 35 U.S.C. 103(a) as being unpatentable over Raymond A. Lorie "Physical Integrity in a Large Segmented Database", ACM Transactions on Database Systems, Vol.2, No. 1., March 1977, pages 91-104.

Claim 55 merely reads on the fact that more than one bit is used to indicate status. Although Lorie does not specifically show more than one bit is used for representing the status of the segment, it would have been obvious to one of ordinary skill in the art to use any number of bits to identify a segment depending on users' requirements.

Regarding claim 56, Lorie discloses an element disposed for selecting storage blocks in response to said one bit and said second bit associated with said selected storage block when Lorie shows that the method selects a free slot (see page 97).

Regarding claim 57, Lorie discloses an element disposed for copying said selected storage blocks to a destination when Lorie shows that a modified segment is copied to a new slot (see page 97).

Regarding claims 58, 61, although Lorie does not specifically show "wherein said destination includes...plurality of file systems", it would have been obvious to one of ordinary skill in the art to include any destination depending on users' requirements.

Claims 74, 75 merely read on the fact that operations of logical sum and difference are performed on storage blocks to form storage image for snapshots. Since the snapshots in the system of Lorie exist as distinct entities, it would have been obvious to one of ordinary skill in the art to form a storage image by performing a logical sum or difference on member storage blocks as claimed.

Regarding claim 81, although Lorie does not specifically disclose that the data structure uses no more than four bytes per storage block, it would have been obvious to one of ordinary skill in the art to do so in order to save memory.

.Regarding claim 82, the claimed first and second snapshots are met when Lorie shows saving new states (see page 99). The claimed step of performing an operation on said snapshots merely reads on the fact that snapshots are combinable to form a storage image. It would have been obvious to one of ordinary skill in the art to do so in order to get an image of a plurality of snapshots.

Claims 84, 85 merely read on the fact that operations of logical sum and difference are performed on snapshots. Since the snapshots in the system of Lorie exist as distinct entities, it would have been obvious to one of ordinary skill in the art to make copies by including or excluding a selected range of snapshots and by copying to a destination in order to use existing snapshots to readily create new ones and save processing time.

Regarding claim 86, official notice is taken that it is well known in the art to copy an image to a tape, a disk, a data structure in a second file system, a set of network messages or a destination distributed over a plurality of file system. Therefore, it would

have been obvious to one of ordinary skill in the art to include the above means as destination in order to copy an image depending on availability of the equipment.

Regarding claims 91, 92, although Lorie does not specifically show the step of optimizing said sequence of member storage blocks for a file system operation in a RAID file system, it is well known in the art to use RAID for backing up files. Since the operation of backing up files consumes time and resources, it would have been obvious to one of ordinary skill in the art to optimize said sequence in order to save processing time backing it up to RAID.

Claim 93 merely reads on the fact that the sequence of member storage blocks is optimized depending on storage medium. Since reading in parallel would speed up the operation, it would have been obvious to one of ordinary skill in the art to include optimizing said sequence in response to a physical location in a storage medium and ordering said sequence so that said member storage blocks are read in parallel in order to speed up the operation and save processing time.

Regarding claims 94, 95, Lorie discloses that the storage image represents a complete file system and changes to a file system when Lorie shows the new consistent state and the previously saved state (see page 96).

Regarding claim 97, although Lorie does not specify selecting in response to an operator command, it would have been obvious to one of ordinary skill in the art to include this feature in order to allow an operator to arbitrarily control the selecting step.

Regarding claim 98, although Lorie does not specify repeating the selection in response to a remote device, it would have been obvious to one of ordinary skill in the art to include this feature in order to allow control of the system remotely.

Regarding claim 99, the claimed incremental mirror copy including a base set of storage blocks stored in a first storage medium merely reads on the original set of storage blocks for the file system of Lorie. Lorie also discloses an incremental storage block when Lorie shows saving a new state (see page 98). Although Lorie does not show that the incremental storage block is in a second storage medium, it would have been obvious to one of ordinary skill in the art to store the incremental set of storage blocks in a second storage medium in order to keep track of the changes separately for easy retrieval.

Claim 100 merely reads on the fact that storage media have different speeds and that snapshots are more recent than the original consistent file system. Official notice is taken that it is well known for storage media to have different speeds. Furthermore, clearly snapshots are more recent than the original data in the file system. Therefore, it would have been obvious to one of ordinary skill in the art to include storing the original base set of storage blocks in slower medium than the incremental set of storage blocks in order to save disk space.

Regarding claim 101, clearly the incremental set of storage blocks is responsive to a plurality of updates of said file system since it stores all changes to the file system.

Claim 102 merely reads on the fact that the system of Lorie continuously creates snapshots of the file system (see the abstract). Since each snapshot includes a set of

storage blocks, it would have been obvious to one of ordinary skill in the art to include a up to date set of storage blocks in the incremental mirror copy in order to keep track of the states of each snapshot.

Claim 103 merely reads on the fact that snapshots can be deleted to free up space. Lorie clearly teaches this feature when Lorie shows that after each save all shadow copies are reclaimed (see page 99, discussion).

Regarding claim 105, although Lorie does not specifically show a storage image as claimed, since snapshots exist as distinct entities, it would have been obvious to one of ordinary skill in the art to perform an operation on at least two of said snapshots in order to benefit from existing snapshot formats.

Regarding claim 106, Lorie discloses an incremental mirror of a file system including a first set of storage blocks forming a copy of a first consistent version of the file system when Lorie shows the creation of the first snapshot. The claimed second set of storage blocks including a set of changes between first and second consistent versions is met when Lorie shows the creation of the next snapshot (see the abstract). Clearly, a complete copy of the file system can be constructed from said first and second sets. Although Lorie does not show that the first and second sets are stored in different media, it would have been obvious to one of ordinary skill in the art to do so in order to store the first version in slow inexpensive medium such as tape and the second versions in a faster medium such as disk for saving cost.

Regarding claim 107, clearly tape can store more data than a disk and is slower than a disk.

Regarding claim 108, clearly the second set of member storage blocks is responsive to a plurality of updates of said file system since it stores all changes to the file system.

Claim 109 merely reads on the fact that the system of Lorie continuously creates snapshots of the file system (see the abstract). Since each snapshot includes a set of storage blocks, it would have been obvious to one of ordinary skill in the art to include a up to date set of storage blocks in the second set of member storage blocks in order to keep track of the states of each snapshot.

Claim 110 merely reads on the fact that snapshots can be deleted to free up space. Lorie clearly teaches this feature when Lorie shows that after each save all shadow copies are reclaimed (see page 99, discussion).

Regarding claim 113, although Lorie does not specifically show an image stream including a set of storage blocks of both first and second snapshots, since snapshots exist as distinct entities, it would have been obvious to one of ordinary skill in the art to include both first and second snapshots as claimed in order to benefit from existing snapshot formats. Furthermore, copying said member storage blocks clearly does not alter the snapshots in any way.

Regarding claims 118, 119 although Lorie does not specifically show that said storage image indicates a logical sum or difference of two sets of member storage blocks, it would have been obvious to one of ordinary skill in the art to use logical operations to perform sums and differences on sets of storage blocks in order to benefit from their existing format.

Regarding claim 122, although Lorie does not specifically show a format for shadow snapshots, it would have been obvious to one of ordinary skill in the art to use a format that facilitates set management operation in order to process them efficiently.

Regarding claim 124, although Lorie does not specify a size for a shadow snapshot, it would have been obvious to one of ordinary skill in the art to use about one byte per storage block in order to save memory.

Regarding claims 135, 145, although Lorie does not specifically show that the data structure uses no more than four bytes per storage block, it would have been obvious to one of ordinary skill in the art to make said mark-on-allocate and said mark-on-deallocate images use no more than four bytes per storage block in order to save memory.

Claims 78, 83, 130, 140, 147 merely read on the well-known fact that set management operation clearly includes logical sum or difference.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Uyen T. Le whose telephone number is 703-305-4134. The examiner can normally be reached on M-F 7:00-5:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Safet Metjahic can be reached on 703-308-1436. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

22 September 2004

Ule

UYEN LE
PRIMARY EXAMINER

103.1068.01 09/854,187

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: Steve KLEIMAN et al.	Art Unit: 2171
Serial No.: 09/854,187	Examiner: Uyen T LE
Filed: May 10, 2001	Tel: (703) 305-4134
For: File System Image Transfer	Office Action Mailed: Sept. 29, 2004

RECEIVED
CENTRAL FAX CENTER
DEC 2 4 2004

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

RESPONSE TO OFFICE ACTION

Dear Examiner:

This paper is responsive to the outstanding Office Action dated as shown above. Applicants conditionally request an extension of time in the event that one is required for this paper. In accordance with 37 CFR § 1.136(a)(3), authorization is hereby granted to charge any required extension of time fees under 37 CFR § 1.17 to Deposit Account No. 50-0365. Authorization is also hereby granted to charge any additional claim fees and any other fees necessary for filing of this paper to Deposit Account No. 50-0365.

103.1068.01 09/854,187

IN THE CLAIMS:

Please amend the claims as follows:

Claims 52 and 53 (Cancelled)

54. (Currently Amended) A file system that stores and retrieves information in ~~,~~ ~~having~~ a plurality of storage blocks, said file system ~~and~~ including a plurality of bits associated with each one of said plurality of storage blocks, at least one of said plurality of bits identifying whether said one storage block was part of said file system at a time earlier than a current consistent version of said file system, and at least a second one of said plurality of bits identifying whether said one storage block was part of said file system at a second time earlier than a current consistent version of said file system.

55. (Cancelled)

56. (Currently Amended) A file system as in claim 54 ~~55~~, including an element disposed for selecting storage blocks in response to said one bit and said second one bit associated with said selected storage blocks.

57. (Previously Presented) A file system as in claim 56, including an element disposed for copying said selected storage blocks to a destination.

58. (Previously Presented) A file system as in claim 57, wherein said destination includes: a tape, a disk, a data structure in a second file system, a set of network messages, or a destination distributed over a plurality of file systems.

59. (Previously Presented) A file system as in claim 54, including an element disposed for selecting storage blocks in response to said one bit associated with said selected storage blocks.

60. (Previously Presented) A file system as in claim 59, including an element disposed for copying said selected storage blocks to a destination.

61. (Previously Presented) A file system as in claim 60, wherein said destination includes: a tape, a disk, a data structure in a second file system, a set of network messages, or a destination distributed over a plurality of file systems.

62. (Currently Amended) A file system having a plurality of storage blocks, said file system including a snapshot including a set of member storage blocks selected from said plurality, said member storage blocks forming a consistent file system other than an active file system, ~~;~~ said

snapshot being disposed as an object in said file system ~~, wherein said file system is responsive to at least one file system request with regard to said snapshot~~.

63. (Previously Presented) A file system as in claim 62, including

a mark on allocate image of a set of member storage blocks selected from said plurality, said member storage blocks having been added to said snapshot; and

a storage image defined based on said snapshot and said mark on allocate image, said storage image indicating a set of member storage blocks selected from said plurality.

64. (Previously Presented) A file system as in claim 63, wherein said storage image is defined with regard to a logical sum operation on said snapshot and said mark on allocate image.

65. (Previously Presented) A file system as in claim 62, including

a mark on deallocate image of a set of member storage blocks selected from said plurality, said member storage blocks having been removed from said snapshot; and

a storage image defined based on said snapshot and said mark on deallocate image, said storage image indicating a set of member storage blocks selected from said plurality.

66. (Previously Presented) A file system as in claim 62, including

a shadow snapshot of a set of member storage blocks selected from said plurality, said member storage blocks having formed a consistent file system other than an active file system, with a set of selected member storage blocks removed from said consistent file system; and

a storage image defined based on said snapshot and said shadow snapshot, said storage image indicating a set of member storage blocks selected from said plurality.

67. (Previously Presented) A file system as in claim 62, including an indicator of which ones of said member storage blocks have been copied.

68. (Previously Presented) A file system as in claim 62, including a plurality of said snapshots; wherein said plurality of said snapshots are associated with an array of bits, said array having one set of bits for each storage block in said plurality of storage blocks, said set of bits having at least one bit for each said snapshot.

69. (Previously Presented) A file system as in claim 62, wherein said file system can manipulate said snapshot without having to traverse a hierarchy of file system objects within said snapshot.

70. (Previously Presented) A file system as in claim 62, wherein said snapshot includes a data structure disposed in a format allowing for a set management operation to be performed efficiently.

71. (Previously Presented) A file system as in claim 62, wherein said snapshot includes an array of bits, said array having one bit for each storage block in said plurality.

72. (Previously Presented) A file system as in claim 62, including

a plurality of said snapshots; and

a storage image determined based on said plurality of snapshots;

said storage image defining a second set of member storage blocks selected from said plurality.

73. (Previously Presented) A file system as in claim 72, including an indicator of which ones of said storage blocks in said storage image have been copied.

74. (Previously Presented) A file system as in claim 72, wherein said storage image is a result of a logical sum or difference performed on said set of member storage blocks for said snapshot and a set of member storage blocks for a second said snapshot.

75. (Previously Presented) A file system as in claim 72, wherein said storage image is a result of a logical sum or difference performed on said set of member storage blocks for said snapshot and a set of member storage blocks for a second said storage image.

76. (Previously Presented) A file system as in claim 72, wherein said storage image is a result of a set management operation on said set of member storage blocks for said snapshot.

77. (Previously Presented) A file system as in claim 62, wherein said snapshot includes a data structure disposed in a format allowing for a set management operation to be performed in O(n) time or less, where n is a number of storage blocks in said plurality, without reading any contents of said storage blocks in said plurality.

78. (Previously Presented) A file system as in claim 77, wherein said set management operation is a logical sum or difference.

79. (Previously Presented) A file system as in claim 62, wherein said snapshot includes a data structure identifying which storage blocks in said plurality are member storage blocks of said snapshot.

80. (Previously Presented) A file system as in claim 79, wherein said data structure uses no more than 1/100th of an amount of storage required by said storage blocks in said plurality.

81. (Previously Presented) A file system as in claim 79, wherein said data structure uses no more than four bytes per storage block in said plurality.

82. (Previously Presented) A method of operating a file server, said method including steps for

forming a first snapshot of a first consistent state of said file system at a selected time, said first snapshot including an indication of a set of storage blocks in said first consistent state;

forming a second snapshot of a second consistent state of said file system, said second snapshot including an indication of a set of storage blocks in said second consistent state; and

performing an operation on said first and second snapshots to form a storage image including an indication of at least some storage blocks in said file system.

83. (Previously Presented) A method as in claim 82, wherein said operation includes a logical sum or difference.

84. (Previously Presented) A method as in claim 82, wherein

said operation includes a logical sum or difference; and

a purpose of said operation includes making a copy including or excluding a selected range of snapshots.

85. (Previously Presented) A method as in claim 82, wherein

said operation includes a logical sum or difference; and

a purpose of said operation includes copying said storage image to a destination.

86. (Previously Presented) A method as in claim 85, wherein said destination includes a tape, a disk, a data structure in a second file system, a set of network messages, or a destination distributed over a plurality of file systems.

87. (Currently Amended) A method to be performed in a file system, said file system having a plurality of storage blocks, said method including steps for

defining a storage image of a set of member storage blocks selected from said plurality, said storage image being formed based on a set of member storage blocks forming a consistent file system other than an active file system; ~~and~~

forming an image stream of a sequence of member storage blocks selected from said storage image; and

sending said image stream from a source file system to a destination file system.

88. (Previously Presented) A method as in claim 87, including steps for associating a block location with each one of said sequence.

89. (Previously Presented) A method as in claim 87, further including steps for reconstructing a file system based on said image stream.

90. (Previously Presented) A method as in claim 87, wherein said steps for forming are performed in response to a selected operation to be performed on said member storage blocks, said selected operation being other than an operation on an active file system

91. (Previously Presented) A method as in claim 87, wherein said steps for forming include steps for optimizing said sequence of member storage blocks for a file system operation.

92. (Previously Presented) A method as in claim 87, wherein said steps for forming include steps for optimizing said sequence of member storage blocks for a file system operation in a RAID file system.

93. (Previously Presented) A method as in claim 87, wherein said steps for forming include steps for

optimizing said sequence of member storage blocks based on a physical location in a storage medium for each said member storage block, said storage medium having a plurality of storage elements capable of being read in parallel; and

ordering said sequence of member storage blocks so that said member storage blocks can be optimally read in parallel from said plurality of storage elements.

94. (Previously Presented) A method as in claim 87, wherein said storage image represents a complete file system.

95. (Previously Presented) A method as in claim 87, wherein said storage image represents a set of changes to a file system.

96. (Previously Presented) A method as in claim 87, including repeating said defining step at periodic intervals.

97. (Previously Presented) A method as in claim 87, including repeating said defining step in response to an operator command.

98. (Previously Presented) A method as in claim 87, including repeating said selecting step in response to a remote device.

Claims 99 to 103 (Cancelled)

104. (Currently Amended) Apparatus including a file system <u>that stores and retrieves information, said file system</u> including a plurality of snapshots thereof, each representing an associated consistent state at an associated selected time; and each said snapshot including an indication of a set of storage blocks in said associated consistent state, said indication being recorded in at least one storage block in said associated consistent state.

105. (Previously Presented) Apparatus as in claim 104, including a storage image defining at least some storage blocks in said file system, said storage image based on an operation on at least two of said snapshots.

Claims 106 to 110 (Cancelled)

111. (Previously Presented) In a file system having a plurality of storage blocks, a data structure including

a first snapshot of a set of member storage blocks selected from said plurality, said member storage blocks forming a consistent file system other than an active file system;

said first snapshot being represented as an object in said file system and having a set of storage blocks for recording said first snapshot;

whereby copying said member storage blocks in said first snapshot has the property of preserving at least one snapshot recorded in said file system at a time of said first snapshot.

112. (Previously Presented) A data structure as in claim 111, including

a second snapshot of a set of member storage blocks selected from said plurality, said member storage blocks forming a consistent file system other than an active file system;

said second snapshot being represented as an object in said file system and having a set of storage blocks for recording said second snapshot;

whereby copying said member storage blocks in said second snapshot has the property of preserving at least one snapshot recorded in said file system at a time of said second snapshot.

113. (Previously Presented) A data structure as in claim 111, including

an image stream including a set of storage blocks including both said first snapshot and said second snapshot;

whereby copying said member storage blocks in said image stream has the property of preserving both said first snapshot and said second snapshot.

114. (Previously Presented) In a file system having a plurality of storage blocks, a data structure including

a snapshot of a set of member storage blocks selected from said plurality, said member storage blocks forming a consistent file system other than an active file system;

said snapshot being represented as an object in said file system and having a set of storage blocks for recording said snapshot;

whereby a backup and restore operation on said file system has the property of preserving said snapshot within said file system.

Claims 115 to 120 (Cancelled)

121. (Currently Amended) A ~~In a~~ file system that stores and retrieves information in ~~having~~ a plurality of storage blocks, including a data structure ~~stored in said file system,~~ including a shadow snapshot of a set of member storage blocks selected from said plurality, said member storage blocks having formed a consistent file system other than an active file system, with a set of selected member storage blocks removed from said consistent file system;

wherein said shadow snapshot is disposed as a single object in said file system, whereby said file system can manipulate said snapshot without having to traverse a hierarchy of file system objects within said snapshot.

122. (Currently Amended) A file system ~~data structure~~ as in claim 121, wherein said shadow snapshot is disposed in a format allowing for a set management operation to be performed efficiently.

123. (Currently Amended) A file system ~~data structure~~ as in claim 121, wherein said shadow snapshot uses, in addition to said member storage blocks, no more than 1/100th of an amount of storage required by said storage blocks in said plurality.

124. (Currently Amended) A file system ~~data structure~~ as in claim 121, wherein said shadow snapshot uses, in addition to said member storage blocks, no more than one byte per storage block in said plurality.

125. (Cancelled)

126. (Currently Amended) A file system ~~data structure~~ as in claim 121, wherein said removed member storage blocks are responsive to completion of a processing operation.

127. (Currently Amended) A file system ~~data structure~~ as in claim 126, wherein said processing operation includes a file system operation.

128. (Currently Amended) A file system ~~data structure~~ as in claim 126, wherein said processing operation includes reuse of said selected member storage blocks by said file system.

129. (Currently Amended) A file system ~~data structure~~ as in claim 121, wherein said shadow snapshot is disposed in a format allowing for a set management operation to be performed in O(n) time or less, where n is a number of storage blocks in said plurality, without reading any contents of said storage blocks in said plurality.

130. (Currently Amended) A file system ~~data structure~~ as in claim 129, wherein said set management operation is a logical sum or difference.

131. (Currently Amended) A ~~In a~~ file system that stores and retrieve information in ~~having~~ a plurality of storage blocks ~~,~~ including a data structure that includes ~~including~~ a mark on

allocate image of a set of member storage blocks selected from said plurality, said member storage blocks having been added to a snapshot that originally formed a consistent file system;

wherein said mark on allocate storage image is disposed as a single object in said file system, whereby said file system can manipulate said snapshot without having to traverse a hierarchy of file system objects within said snapshot.

132. (Cancelled)

133. (Currently Amended) A file system ~~data structure~~ as in claim 131, wherein said mark on allocate image is disposed in a format allowing for a set management operation to be performed efficiently.

134. (Currently Amended) A file system ~~data structure~~ as in claim 131, wherein said mark on allocate storage image uses no more than 1/100th of an amount of storage required by said storage blocks in said plurality.

135. (Currently Amended) A file system ~~data structure~~ as in claim 131, wherein said mark on allocate image uses no more than four bytes per storage block in said plurality.

136. (Currently Amended) A file system ~~data structure~~ as in claim 131, said member storage blocks having been selected responsive to completion of a processing operation.

137. (Currently Amended) A file system ~~data structure~~ as in claim 136, wherein said processing operation includes a file system operation.

138. (Currently Amended) A file system ~~data structure~~ as in claim 136, wherein said processing operation includes reuse of said selected member storage blocks by said file system.

139. (Currently Amended) A file system ~~data structure~~ as in claim 131, wherein said mark on allocate image is disposed in a format allowing for a set management operation to be performed in O(n) time or less, where n is a number of storage blocks in said plurality, without reading any contents of said storage blocks in said plurality.

140. (Currently Amended) A file system ~~data structure~~ as in claim 139, wherein said set management operation is a logical sum or difference.

141. (Currently Amended) A ~~In a~~ file system that stores and retrieve information in ~~having~~ a plurality of storage blocks ~~,~~ including a data structure that includes ~~stored in said file system, including~~ a mark on deallocate image of a set of member storage blocks selected from said plurality, said member storage blocks having been removed from a snapshot that originally formed a consistent file system;

wherein said mark on deallocate storage image is disposed as a single object in said file system, whereby said file system can manipulate said snapshot without having to traverse a hierarchy of file system objects within said snapshot.

142 (Cancelled)

143. (Currently Amended) A file system ~~data structure~~ as in claim 141, wherein said mark on deallocate image is disposed in a format allowing for a set management operation to be performed efficiently.

144. (Currently Amended) A file system ~~data structure~~ as in claim 141, wherein said mark on deallocate storage image uses no more than 1/100th of an amount of storage required by said storage blocks in said plurality.

145. (Currently Amended) A file system ~~data structure~~ as in claim 141, wherein said mark on deallocate image uses no more than four bytes per storage block in said plurality.

146. (Currently Amended) A file system ~~data structure~~ as in claim 141, wherein said mark on deallocate image is disposed in a format allowing for a set management operation to be performed in O(n) time or less, where n is a number of storage blocks in said plurality, without reading any contents of said storage blocks in said plurality.

147. (Currently Amended) A <u>file system</u> ~~data structure~~ as in claim 146, wherein said set management operation is a logical sum or difference.

REMARKS:

Status

After this response, claims 54, 56 to 98, 104, 105, 111 to 114, 121 to 124, 126 to131, 133 to 141, and 143 to 147 are pending. Claims 54, 56, 62, 87, 104, 121 to 124, 126 to 131, 133 to 141, and 143 to 147 have been amended, and claims 52, 53, 55, 99 to 103, 106 to 110, 115 to 120, 125, 132 and 142 have been cancelled. Claims 54, 62, 82, 87, 104, 111, 114, 121, 131 and 141 are the independent claims. Reconsideration and further examination are respectfully requested.

Section 101 Rejection

Claims 54 to 61, 99 to 105, and 115 to 147 were rejected under 35 U.S.C. § 101. Of these, claims 54, 56 to 61, 104, 105, 121 to 124, 126 to 131, 133 to 141, and 143 to 147 are still pending. These claims have been amended to recite "[a]file system that stores and retrieves information." Applicants submit that such a file system is clearly statutory subject matter. Accordingly, withdrawal is respectfully requested of the rejection under § 101.

Section 102 and 103 rejections

All pending claims were rejected under 35 U.S.C. § 102 or § 103 over Raymond A. Lorie, "Physical Integrity in a Large Segmented Database," ACM Transactions on Database Systems, vol. 2, no. 1, Mar. 1997, pp 91-104 (Lorie), except for claims 64, 117, and 143. Applicants note that the language of these claims substantially matches language in claims that were rejected and also

note that all claims are listed as rejected in the Disposition section of the Office Action Summary. Applicants have therefore proceeded on the basis that all claims were rejected under § 102 or § 103 over Lorie.

The claims are discussed below, grouped by independent claim.

<u>Claim 52</u>: This claim has been cancelled, rendering the rejection thereof moot.

<u>Claim 53</u>: This claim has been cancelled, rendering the rejection thereof moot.

<u>Claims 54 and 56 to 61</u>: Independent claim 54 is reproduced below:

> 54. (Currently Amended) A file system that stores and retrieves information in a plurality of storage blocks, said file system including a plurality of bits associated with each one of said plurality of storage blocks, at least one of said plurality of bits identifying whether said one storage block was part of said file system at a time earlier than a current consistent version of said file system, and at least a second one of said plurality of bits identifying whether said one storage block was part of said file system at a second time earlier than a current consistent version of said file system.

Lorie is not seen by Applicants to disclose or to suggest the foregoing features of claim 54, at least with respect to "identifying whether said one storage block was part of said file system at a second time earlier than a current consistent version of said file system" in combination with "identifying whether said one storage block was part of said file system at a time earlier than a current consistent version of said file system."

Before this response, this feature was recited by claim 55. In the § 103 rejection of that claim, the Office Action stated the following:

> Claim 55 merely reads on the fact that more than one bit is used to indicate status. Although Lorie does not specifically show more than one bit is used for representing the status of the segment, it would have been obvious

to one of ordinary skill in the art to use any number of bits to identify a segment depending on users' requirements.

Lorie does not teach "identifying whether said one storage block was part of said file system *at a second time* earlier than a current consistent version of said file system" (emphasis added). Applicants further submit that the following text from section 3 on page 96 of Lorie teaches directly against this feature:

> Let us suppose S_k is in a state of integrity (D_k) at time t and that a series of changes are made to S_k between time t and time $t + \Delta t$, at which time we decide to permanently store the new state D_k' onto disk. If a system failure occurs between t and $t + \Delta t$, we want to be able to return to the state D_k. If no failure occurs in that interval, the new state D_k' is recorded and D_k is forgotten.

In particular, Applicants submit that teaching that state "D_k is forgotten" is counter to a teaching of "identifying whether said one storage block was part of said file system *at a second time* earlier than a current consistent version of said file system" in addition to "identifying whether said one storage block was part of said file system at a time earlier than a current consistent version of said file system," as recited by claim 54.

For at least the foregoing reasons, claim 54 and its dependent claims are believed to be allowable over Lorie. Reconsideration and withdrawal are therefore requested of the rejections of those claims.

<u>Claims 62 to 81</u>: Independent claim 62 is reproduced below:

> 62. (Currently Amended) A file system having a plurality of storage blocks, said file system including a snapshot including a set of member storage blocks selected from said plurality, said member storage blocks forming a consistent file system other than an active file system, said snapshot being disposed as an object in said file system.

Lorie is not seen by Applicants to disclose or to suggest the foregoing features of claim 62, at least with respect to "said snapshot being disposed as an object in said file system."

Lorie concerns a database system. The Office Action apparently has broadly read the database system to be equivalent to the claimed file system. The Office Action also has apparently read Lorie's "state of integrity" to be equivalent to the claimed snapshot. Applicants respectfully submit that given these equivalences, Lorie would have to teach that its "state of integrity" is stored as an object in Lorie's database in order to teach the claimed feature of "said snapshot being disposed as an object in said file system." Applicants see no such teaching in Lorie.

For at least the foregoing reasons, claim 62 and its dependent claims are believed to be allowable over Lorie. Reconsideration and withdrawal are therefore requested of the rejections of those claims.

Claims 82 to 86: Independent claim 82 is reproduced below:

> 82. (Previously Presented) A method of operating a file server, said method including steps for
>
> forming a first snapshot of a first consistent state of said file system at a selected time, said first snapshot including an indication of a set of storage blocks in said first consistent state;
>
> forming a second snapshot of a second consistent state of said file system, said second snapshot including an indication of a set of storage blocks in said second consistent state; and
>
> performing an operation on said first and second snapshots to form a storage image including an indication of at least some storage blocks in said file system.

Lorie is not seen by Applicants to disclose or to suggest the foregoing features of claim 82, at least with respect to "performing an operation on said first and second snapshots to form a storage image including an indication of at least some storage blocks in said file system."

Applicants submit that in order for such an operation to be performed, both snapshots must be present. However, Lorie's teaching that state "D_k is forgotten" appears to Applicants to teach against having both a first and a second snapshot present.

For at least the foregoing reasons, claim 82 and its dependent claims are believed to be allowable over Lorie. Reconsideration and withdrawal are therefore requested of the rejections of those claims.

Claims 87 to 98: Independent claim 87 is reproduced below:

> 87. (Currently Amended) A method to be performed in a file system, said file system having a plurality of storage blocks, said method including steps for
>
> defining a storage image of a set of member storage blocks selected from said plurality, said storage image being formed based on a set of member storage blocks forming a consistent file system other than an active file system;
>
> forming an image stream of a sequence of member storage blocks selected from said storage image; and
>
> sending said image stream from a source file system to a destination file system.

Lorie is not seen by Applicants to disclose or to suggest the foregoing features of claim 87, at least with respect to "sending said image stream from a source file system to a destination file system."

Lorie concerns a database system. The Office Action apparently has broadly read the database system to be equivalent to the claimed file system. Applicants respectfully submit that given this equivalence, Lorie would have to teach sending an image stream from a source database to a destination database in order to teach "sending said image stream from a source file system to a destination file system." Applicants do not understand Lorie to be concerned with such a process.

For at least the foregoing reasons, claim 87 and its dependent claims are believed to be allowable over Lorie. Reconsideration and withdrawal are therefore requested of the rejections of those claims.

Claims 99 to 103: These claims have been cancelled, rendering the rejections thereof moot.

Claims 104 and 105: Independent claim 104 is reproduced below:

> 104. (Currently Amended) Apparatus including a file system that stores and retrieves information, said file system including a plurality of snapshots thereof, each representing an associated consistent state at an associated selected time; and each said snapshot including an indication of a set of storage blocks in said associated consistent state, said indication being recorded in at least one storage block in said associated consistent state.

Lorie is not seen by Applicants to disclose or to suggest the foregoing features of claim 104, at least with respect to "said file system including a plurality of snapshots thereof, each representing an associated consistent state at an associated selected time." In particular, Lorie's teaching that state "D_k is forgotten" appears to Applicants to teach against having a plurality of snapshots.

For at least the foregoing reasons, claim 104 and its dependent claim are believed to be allowable over Lorie. Reconsideration and withdrawal are therefore requested of the rejections of those claims.

Claims 106 to 110: These claims have been cancelled, rendering the rejections thereof moot.

Claims 111 to 113: Independent claim 111 is reproduced below.

111. (Previously Presented) In a file system having a plurality of storage blocks, a data structure including

a first snapshot of a set of member storage blocks selected from said plurality, said member storage blocks forming a consistent file system other than an active file system;

said first snapshot being represented as an object in said file system and having a set of storage blocks for recording said first snapshot;

whereby copying said member storage blocks in said first snapshot has the property of preserving at least one snapshot recorded in said file system at a time of said first snapshot.

Lorie is not seen by Applicants to disclose or to suggest the foregoing features of claim 111, at least with respect to "said first snapshot being represented as an object in said file system."

Lorie concerns a database system. The Office Action apparently has broadly read the database system to be equivalent to the claimed file system. The Office Action apparently has also read Lorie's "state of integrity" to be equivalent to the claimed snapshot. Applicants respectfully submit that given these equivalences, Lorie would have to teach that its "state of integrity" is represented as an object in Lorie's database in order to teach the claimed feature of "said first snapshot being represented as an object in said file system." Applicants see no such teaching in Lorie.

For at least the foregoing reasons, claim 111 and its dependent claims are believed to be allowable over Lorie. Reconsideration and withdrawal are therefore requested of the rejections of those claims.

Claim 114: Independent claim 114 is reproduced below.

114. (Previously Presented) In a file system having a plurality of storage blocks, a data structure including

a snapshot of a set of member storage blocks selected from said plurality, said member storage blocks forming a consistent file system other than an active file system;

said snapshot being represented as an object in said file system and having a set of storage blocks for recording said snapshot;

whereby a backup and restore operation on said file system has the property of preserving said snapshot within said file system.

Lorie is not seen by Applicants to disclose or to suggest the foregoing features of claim 114, at least with respect to "said snapshot being represented as an object in said file system."

Lorie concerns a database system. The Office Action apparently has broadly read the database system to be equivalent to the claimed file system. The Office Action apparently has also read Lorie's "state of integrity" to be equivalent to the claimed snapshot. Applicants respectfully submit that given these equivalences, Lorie would have to teach that its "state of integrity" is represented as an object in Lorie's database in order to teach the claimed feature of "said snapshot being represented as an object in said file system." Applicants see no such teaching in Lorie.

For at least the foregoing reasons, claim 114 is believed to be allowable over Lorie. Reconsideration and withdrawal are therefore requested of the rejections of this claim.

Claims 115 to 120: These claims have been rejected, rending the rejections thereof moot.

Claims 121 to 130: Claim 121 is reproduced below.

121. (Currently Amended) A file system that stores and retrieves information in a plurality of storage blocks, including a data structure including a shadow snapshot of a set of member storage blocks selected from said plurality, said member storage blocks having formed a consistent file system other than an active file system, with a set of selected member storage blocks removed from said consistent file system;

wherein said shadow snapshot is disposed as a single object in said file system, whereby said file system can manipulate said snapshot without having to traverse a hierarchy of file system objects within said snapshot.

Lorie is not seen by Applicants to disclose or to suggest the foregoing features of claim 121, at least with respect to "said shadow snapshot is disposed as a single object in said file system."

Lorie concerns a database system. The Office Action apparently has broadly read the database system to be equivalent to the claimed file system. Applicants are unsure of exactly what feature of Lorie's database system is being equated with the claimed shadow snapshot. Nonetheless, Applicants certainly do not see any teaching of disposing a shadow snapshot or its equivalent as a single object in Lorie's database. Thus, Lorie is not seen by Applicants to teach that "said shadow snapshot is disposed as a single object in said file system."

For at least the foregoing reasons, claim 121 and its dependent claims are believed to be allowable over Lorie. Reconsideration and withdrawal are therefore requested of the rejections of those claims.

<u>Claims 131 to 140</u>. Claim 131 is reproduced below.

131. (Currently Amended) A file system that stores and retrieve information in a plurality of storage blocks including a data structure that includes a mark on allocate image of a set of member storage blocks selected from said plurality, said member storage blocks having been added to a snapshot that originally formed a consistent file system;

wherein said mark on allocate storage image is disposed as a single object in said file system, whereby said file system can manipulate said snapshot without having to traverse a hierarchy of file system objects within said snapshot.

Lorie is not seen by Applicants to disclose or to suggest the foregoing features of claim 131, at least with respect to "said mark on allocate storage image is disposed as a single object in said file system."

Lorie concerns a database system. The Office Action apparently has broadly read the database system to be equivalent to the claimed file system. The Office Action apparently has also read Lorie's "state of integrity" to be equivalent to the claimed storage image. Applicants respectfully submit that given these equivalences, Lorie would have to teach that its "state of integrity" is disposed as a single object in Lorie's database in order to teach the claimed feature that "said mark on allocate storage image is disposed as a single object in said file system." Applicants see no such teaching in Lorie.

For at least the foregoing reasons, claim 131 and its dependent claims are believed to be allowable over Lorie. Reconsideration and withdrawal are therefore requested of the rejections of those claims.

Claims 141 to 147: Claim 141 is reproduced below.

> 141. (Currently Amended) A file system that stores and retrieve information in a plurality of storage blocks including a data structure that includes a mark on deallocate image of a set of member storage blocks selected from said plurality, said member storage blocks having been removed from a snapshot that originally formed a consistent file system;
>
> wherein said mark on deallocate storage image is disposed as a single object in said file system, whereby said file system can manipulate said snapshot without having to traverse a hierarchy of file system objects within said snapshot.

Lorie is not seen by Applicants to disclose or to suggest the foregoing features of claim 141, at least with respect to "said mark on deallocate storage image is disposed as a single object in said file system."

Lorie concerns a database system. The Office Action apparently has broadly read the database system to be equivalent to the claimed file system. The Office Action apparently has also read Lorie's "state of integrity" to be equivalent to the claimed storage image. Applicants respectfully submit that given these equivalences, Lorie would have to teach that its "state of integrity" is disposed as a single object in Lorie's database in order to teach the claimed feature that "said mark on deallocate storage image is disposed as a single object in said file system." Applicants see no such teaching in Lorie.

For at least the foregoing reasons, claim 141 and its dependent claims are believed to be allowable over Lorie. Reconsideration and withdrawal are therefore requested of the rejections of those claims.

Closing

In view of the foregoing amendments and remarks, the entire application is believed to be in condition for allowance, and such action is respectfully requested at the Examiner's earliest convenience.

Applicants' undersigned attorney can be reached at (614) 486-3585. All correspondence should continue to be directed to the address indicated below.

Respectfully submitted,

Dane C. Butzer

Dated: December 24, 2004

Dane C. Butzer
Reg. No. 43,521

Swernofsky Law Group PC
P.O. Box 390013
Mountain View, CA 94039-0013
(650) 947-0700

INFORMATION DISCLOSURE STATEMENT	Applicant(s):	David Hitz
	US Patent No.	5,819,292
	Issued:	October 6, 1998
	For:	Method for Maintaining Consistent States of a File System and for Creating User-Accessible Read-Only Copies of a File System
	Group Art Unit:	N/A
	Examiner:	N/A
	Attorney Docket No.:	347155-29

Commissioner of Patents and Trademarks
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

In accordance with the provisions of 37 C.F.R. § 1.56(a) and 37 C.F.R. § 1.97, Applicant(s) hereby make of record the references listed on the accompanying Form PTO-1449 for consideration by the Examiner in connection with the examination of the above-identified patent application.

This Information Disclosure Statement:

(a)	☐	accompanies a new patent application submitted herewith.
(b)	☐	is filed within three (3) months of the Filing Date or before the mailing date of a First Office Action on the merits; OR
(c)	☐	after the period defined in (b) but before the mailing date of a Final Rejection or Notice of Allowance, OR
(d)	☐	is filed after the first Office Action and more than three months after the application's filing date or PCT national stage date of entry filing but, as far as is known to the undersigned prior to the mailing date of either a final rejection or a notice of allowance, and is accompanied by either the fee ($180) set forth in 37 CFR § 1.17(p) or a certification as specified in 37 CFR § 1.97(e), as checked below OR
(e)	☐	is filed after the mailing date of either a final rejection or a notice of allowance, and the issue fee has not been paid, and is accompanied by the requisite petition fee ($130) set forth in 37 CFR § 1.17(I)(1) and a certification as specified in 37 CFR § 1.97(e), as checked below. This document is to be considered as a petition requesting consideration of the information disclosure statement.

As required under § 1.97(e), Applicants, through the undersigned, hereby state either that [check the appropriate space]:

(f)	☐	Each item of information contained in the Information Disclosure Statement was first cited in a communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing date of the Information Disclosure Statement; or
(g)	☐	No item of information contained in the Information Disclosure Statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing this Statement after making reasonable inquiry, no item of information contained in the Information Disclosure Statement was known to any individual designated in § 1.56(c) more than three months prior to the filing of the Information Disclosure Statement.

Regarding the concise statements required for any foreign-language references, a copy of an associated foreign Office Action indicating the relevance is provided in English; see MPEP 609.04(a)(III).

It is respectfully requested that each of the references shown on the attached Form PTO-1449 be made of record in this application. Copies of the references are enclosed.

The Commissioner is authorized to charge any deficiencies and credit any overpayment of fees to our Deposit Account No. **07-1896**.

Respectfully submitted,

Date: October 25, 2007

DLA PIPER US LLP

By: Ronald L. Yin

Ronald L. Yin
Reg. No. 27,607
Attorneys for Applicant(s)
2000 University Avenue
East Palo Alto, CA 94303-2248
650-833-2437

☒ *Customer Number or Bar Code Label*

26379
(Insert Customer No. or Attach bar code label here)

EXPRESS MAIL NUMBER: EV 978 428 074 US DATE OF DEPOSIT: October 25, 2007

I hereby certify that this paper is being deposited with the United States Postal Service "EXPRESS MAIL Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to: Mail Stop PATENT APPLICATION, P.O. Box 1450, Alexandria, VA 22313-1450.

Ronald L. Yin

Ronald L. Yin

EM\7223576.1
347155-29

Form PTO-1449 U.S. DEPT. OF COMMERCE Patent and Trademark Office **INFORMATION DISCLOSURE CITATION** (Use several sheets if necessary)	Attorney Docket Number: **347155-29**	U.S. Patent Number: **5,819,292**
	Applicant: **David Hitz**	
	Filing date: **May 31, 1995**	Group art unit: **N/A**

U.S. PATENT DOCUMENTS

Examiner Initial	Patent No. Publication No.	Date	Name	Class	Sub-class	Filing date if appropriate
	5,129,085	July 7, 1992	Yamasaki, Akiko			

OTHER DOCUMENTS (Including Author, Title, Date, Pertinent Pages, Etc.)

	ANITA BORG, et al., "Fault Tolerance Under UNIX", ACM Transactions on Computer Systems, February 1989, pages 1-24, Vol. 7, No. 1. ("Borg").
	MENDEL ROSENBLUM, et al., "The Design and Implementation of a Log-Structured File System", Proceedings of the 13th ACM Symposium on Operating Systems Principles, 1991, pages 1-15. ("Rosenblum I").
	MENDEL ROSENBLUM, et al., "The LFS Storage Manager", USENIX Technical Conference, June 1990, Anaheim, California. ("Rosenblum II").
	RAYMOND A. LORIE, "Physical Integrity in a Large Segmented Database", ACM Transactions on Database Systems, March 1997, pages 91-104, Vol. 2, No. 1. ("Lorie").
	MATTHEW S. HECHT, et al., "Shadow Management of Free Disk Pages With a Linked List", ACM Transactions on Database Systems, December 1983, pages 503-514, Vol. 8, No. 4. ("Hecht").

Examiner:	Date Considered:
EXAMINER: Initial if citation considered, whether or not citation is in conformance with MPEP '609; Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to the applicant.	

database applications with very large files it is generally acknowledged [1, 2] that one should use extents and update-in-place. For *multiversion* database applications, however, shadowing is popular.

ACKNOWLEDGMENT

We gratefully acknowledge helpful discussions and comments on the ideas in this paper by J. Robert Ensor, Thomas B. London, and an anonymous referee.

REFERENCES

1. GRAY, J. Notes on database operating systems. In *Operating Systems—An Advanced Course*. G. Seegmuller, Ed., Springer-Verlag, New York, 1978, pp. 393–481.
2. GRAY, J., MCJONES, P., BLASGEN, M., LINDSAY, B., LORIE, R., PRICE, T., PUTZOLU, F., AND TRAIGER, I. The recovery manager of the system R database manager. *ACM Comput. Surv. 13*, 2 (June 1981), 223–242.
3. HIRSCHBERG, D.S. A class of dynamic memory allocation algorithms. *Commun. ACM 16*, 10 (Oct. 1973), 615–618.
4. KNOWLTON, K.C. A fast storage allocator. *Commun. ACM 8*, 10 (Oct. 1965), 623–625.
5. KNOWLTON, K.C. A programmer's description of L6. *Commun. ACM 9*, 8 (Aug. 1966), 616–625.
6. KNUTH, D.E. *The Art of Computer Programming, Vol. 1: Fundamental Algorithms*, 2nd ed., Addison-Wesley, Reading, Mass., 1975.
7. LORIE, R.A. Physical integrity in a large segmented database. *ACM Trans. Database Syst. 2*, 1 (March 1977), 91–104.
8. MITCHELL, J.G., AND DION, J. A comparison of two network-based file servers. In *Proc. 8th Symp. Operating Systems Principles*, (Pacific Grove, Calif., Dec. 1981), ACM, New York, pp. 45–46.
9. STURGIS, H.E., MITCHELL, J.G., AND ISRAEL, J. Issues in the design and use of a distributed file system. *Oper. Syst. Rev. 14*, 3 (July 1980), 55–69.
10. VERHOFSTAD, J.S.M. Recovery techniques for database systems. *ACM Comput. Surv. 10*, 2 (June 1978), 167–195.

Received April 1982; revised November 1982; accepted December 1982